WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Monday, September 13, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L17	L16 or l15	9
☐	L16	L13 and (overload or overloaded or overloading)	2
☐	L15	L12 and (overload or overloaded or overloading)	7
☐	L14	L13 and l12	0
☐	L13	20001122	121
☐	L12	20001122	33
☐	L11	(variable adj3 (size or sized) adj3 (buffer or queue)) near8 (call or request)	4
☐	L10	(variable adj3 (size or sized) adj3 (buffer or queue)) near8 (overload or overloading)	0
☐	L9	(variable adj3 (size or sized) adj3 (buffer or queue)) near8 (threshold or limit)	2
☐	L8	variable adj3 (size or sized) adj3 (buffer or queue)	219
☐	L7	L5 and l2	2
☐	L6	L5 and l1	103
☐	L5	(throttle or throttling) near8 request	693
☐	L4	(throttle or throttling) near8 request near8 (reduce or reducing) near8 (time) near8 delay	0
☐	L3	((throttle or throttling) near8 (buffer or queue) near8 (control or controlling or management or manage)) same request same delay	0
☐	L2	((buffer or queue) near8 (control or controlling or management or manage)) same request same delay	547
☐	L1	(buffer or queue) near8 (control or controlling or management or manage)	123115

END OF SEARCH HISTORY

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L7: Entry 2 of 2 File: USPT Aug 25, 1998

DOCUMENT-IDENTIFIER: US 5799002 A
TITLE: Adaptive bandwidth throttling for network services

Brief Summary Text (4):
In a host computer system, there can be a plurality of network servers, each of which functions to process requests received from processes executing on remote computer systems (clients). The host computer system is interconnected with the plurality of remote computer systems through the medium of a network, such as Internet. The network typically has a predefined bandwidth capacity which is shared among a number of interconnected computer systems. Each computer system is typically allotted a limited fixed allocation of network bandwidth through which to serve requests received from the client processes. When the allocated network bandwidth becomes saturated with client requests and server responses, some of the data is either delayed in transmission or not delivered to the intended destination. Therefore, some form of request throttling mechanism is necessary to minimize network congestion and efficiently utilize the allocated network bandwidth.

Brief Summary Text (9):
U.S. Pat. No. 5,359,320 discloses a scheduling mechanism for a network arbitration circuit in a broadcast network environment. The scheduling mechanism delays the arbitration circuit from seeking access to the network if the network traffic exceeds a first predetermined threshold and the local traffic in the node exceeds a second predetermined threshold. This scheduling mechanism therefore responds to both local and global congestion to throttle the production of new requests.

Drawing Description Text (3):
FIG. 2 illustrates in flow diagram form the operation of the overall system, including the adaptive bandwidth throttling system, in responding to service requests;

Detailed Description Text (6):
FIG. 1 illustrates in block diagram form the overall architecture of the adaptive bandwidth throttling system BT and an environment in which it operates, while FIGS. 2 and 3 illustrate in flow diagram form the operation of the adaptive bandwidth throttling system BT. The adaptive bandwidth throttling system of the present invention is described as implemented in software, although this system can alternatively be implemented as hardware elements or a combination of hardware and software elements. The adaptive bandwidth throttling system functions to limit the bandwidth usage of each of the plurality of services provided by the plurality of network servers NS1-NSm extant on the host computer system P to the allocated maximum network bandwidth. This control is architected to achieve the minimum impact on the network servers NS1-NSm while concurrently having the maximum impact on network congestion. In selecting an implementation of a bandwidth control mechanism, it is important to note that once a network service initiates a response process, that effort is lost if the response is not executed to completion. Therefore, bandwidth throttling procedures should terminate a service before substantial processing effort is expended. In addition, the typical client-server interaction operates on a request-response paradigm. In particular, the client

transmits a request to the server and the only communication that is received by the client is the response to the request. There is no interprocess communication. Thus, rejecting and/or delaying requests typically results in a subsequent retry by the requesting client process, which consumes both processor and network resources, although in a time delayed manner. The client process can continue to send requests to an overloaded server without the server being capable of throttling this request process. An alternative interprocess communication scheme enables the server to notify the client of the server condition, thereby providing feedback to the requesting client to terminate future requests until the overload is cleared. The adaptive bandwidth throttling system of the present invention is operable in both of these environments.

Detailed Description Text (10):
If the requested operation is permitted, the asynchronous thread queue ATQ enables the operation to execute. If the requested operation is not permitted, the asynchronous thread queue ATQ regulates the operation pursuant to the control procedure indicated by the bandwidth throttling system BT. In particular, operations that are designated as rejected are prevented from proceeding and a control packet can be returned to the requesting client process CP via the network N to indicate that the requested service is unavailable at this time. If the requested operation is designated in the delay category, the asynchronous thread queue ATO stores the received request and returns a control packet to the requesting client process CP to indicate that the operation is pending.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L17: Entry 1 of 9 File: USPT Jul 6, 2004

DOCUMENT-IDENTIFIER: US 6760782 B1
TITLE: Apparatus for controlling internetwork communications

Application Filing Date (1):
20000804

Detailed Description Text (8):
The web server 30 is shown in greater detail in FIG. 3. Various components provide the required connectivity to perform its functionality. A real time operating system 44 controls the interaction between the components. The operating system 44 allocates central processor (CPU) 46 to various tasks, provides memory management, and provides a set of message services and signal services. The message and signal services allow for communication between tasks, and between drivers and a task. Connection to the TCP/IP network 42 Is through an Ethernet driver 48 which transmits and receives messages over Ethernet via an Ethernet communication chip such as an AM79C961. The web server will have an unique global address 18, allowing it to be addressed by other devices on the network. Communication can be over a fiber optic cable or a twisted wire pair. The Ethernet driver 48 manages transmit 50 and receive 51 buffers in memory 52, and interfaces with the AM79C961Ethernet chip. The transmit 50 and receive 51 buffers are shared both by the AM79C961and the Ethernet driver 48. The Ethernet driver 48 also provides a transmit request interface, and a receive indication interface to a TCP/IP stack 54. The AM79C961provides a transmit queue interface, a receive queue interface, and generates interrupts on completion of transmitting a message, and on receiving a new message. The Ethernet driver 46 places receive buffers In the receive queue. In the interrupt routine, the Ethernet driver 46 examines the receive queue. If any messages are in the receive queue, it passes the receive buffer to the TCP/IP stack 54. The TCP/IP stack 54 copies the buffer, and sometime later calls the Ethernet driver 48 to return the buffer and place the returned buffer back into the receive queue.

Detailed Description Text (42):
The maximum size of an individual message is limited to approximately 250 bytes. When the overhead of an Ethernet/TCP header is added, the result is still limited to about 330 bytes- On the 10 Mbps Ethernet, such a message has a transmission time of 270 usec. This means that it is possible to reduce the impact of unsolicited traffic on the I/O scan to less than 500 usec by the simple expedient of throttling the rate at which such requests are accepted. The proxy 116 and bridge 108 mechanisms will do just that. They are almost always stateless: If a message needs to be repeated for any reason, the response may be generated from scratch with no loss of functionality. This in turn reduces, the need for buffer memory space and Improves the latency of data being transmitted. In particular, it makes possible a slave engine which requires very little CPU resources yet can achieve response times in the submillisecond range.

Detailed Description Text (53):
A mimic page which represents some of the hardware physically connected to a programmable logic controller system can be constructed utilizing various graphical programs readily available and that are not an object of the present invention. The

present invention allows a user at a remote location, using a browser, to view the mimic page and actually control various components illustrated in the mimic page. FIG. 4 shows a simple motor start-stop control in ladder logic diagram form that could be available as a mimic page to the user. Pushing a motor start push button 150 will cause a motor start relay 152 to energize through a normally closed stop push button 154 and a normally closed overload contact 156. Auxiliary motor start contact 158 will latch relay 152 after the start push button 150 is released and pilot light 160 will illuminate. Auxiliary motor start, contact 162 will provide power to pump motor 164 which will remain running until stop push button 154 is depressed or overload relay 166 detects an overload condition. In this example, start push button 150, stop push button 154, overload contact 156, auxiliary motor start contacts 158 and 162, and overload relay 166 are inputs to the programmable logic controller system. Relay 152, pilot light 160, and pump motor 164 are outputs. The PLC will have the registers containing the animation data for the inputs and outputs. An application program in the PLC will respond to the inputs to control the outputs.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L17: Entry 3 of 9 File: USPT Nov 20, 2001

DOCUMENT-IDENTIFIER: US 6321272 B1
TITLE: Apparatus for controlling internetwork communications

Application Filing Date (1):
19970910

Detailed Description Text (8):
The web server 30 is shown in greater detail in FIG. 3. Various components provide the required connectivity to perform its functionality. A real time operating system 44 controls the interaction between the components. The operating system 44 allocates central processor (CPU) 46 to various tasks, provides memory management, and provides a set of message services and signal services. The message and signal services allow for communication between tasks, and between drivers and a task. Connection to the TCP/IP network 42 is through an Ethernet driver 48 which transmits and receives messages over Ethernet via an Ethernet communication chip such as an AM79C961. The web server will have an unique global address 18, allowing it to be addressed by other devices on the network. Communication can be over a fiber optic cable or a twisted wire pair. The Ethernet driver 48 manages transmit 50 and receive 51 buffers in memory 52, and interfaces with the AM79C961 Ethernet chip. The transmit 50 and receive 51 buffers are shared both by the AM79C961 and the Ethernet driver 48. The Ethernet driver 48 also provides a transmit request interface, and a receive indication interface to a TCP/IP stack 54. The AM79C961 provides a transmit queue interface, a receive queue interface, and generates interrupts on completion of transmitting a message, and on receiving a new message. The Ethernet driver 46 places receive buffers in the receive queue. In the interrupt routine, the Ethernet driver 46 examines the receive queue. If any messages are in the receive queue, it passes the receive buffer to the TCP/IP stack 54. The TCP/IP stack 54 copies the buffer, and sometime later calls the Ethernet driver 48 to return the buffer and place the returned buffer back into the receive queue.

Detailed Description Text (43):
The maximum size of an individual message is limited is to approximately 250 bytes. When the overhead of an Ethernet/TCP header is added, the result is still limited to about 330 bytes. On the 10 Mbps Ethernet, such a message has a transmission time of 270 usec. This means that it is possible to reduce the impact of unsolicited traffic on the I/O scan to less than 500 usec by the simple expedient of throttling the rate at which such requests are accepted. The proxy 116 and bridge 108 mechanisms will do just that. They are almost always stateless. If a message needs to be repeated for any reason, the response may be generated from scratch with no loss of functionality. This in turn reduces the need for buffer memory space and improves the latency of data being transmitted. In particular, it makes possible a slave engine which requires very little CPU resources yet can achieve response times in the sub-millisecond range.

Detailed Description Text (54):
A mimic page which represents some of the hardware physically connected to a programmable logic controller system can be constructed utilizing various graphical programs readily available and that are not an object of the present invention. The

present invention allows a user at a remote location, using a browser, to view the mimic page and actually control various components illustrated in the mimic page. FIG. 4 shows a simple motor start-stop control in ladder logic diagram form that could be available as a mimic page to the user. Pushing a motor start push button 150 will cause a motor start relay 152 to energize through a normally closed stop push button 154 and a normally closed overload contact 156. Auxiliary motor start contact 158 will latch relay 152 after the start push button 150 is released and pilot light 160 will illuminate. Auxiliary motor start contact 162 will provide power to pump motor 164 which will remain running until stop push button 154 is depressed or overload relay 166 detects an overload condition. In this example, start push button 150, stop push button 154, overload contact 156, auxiliary motor start contacts 158 and 162, and overload relay 166 are inputs to the programmable logic controller system. Relay 152, pilot light 160, and pump motor 164 are outputs. The PLC will have the registers containing the animation data for the inputs and outputs. An application program in the PLC will respond to the inputs to control the outputs.

Previous Doc Next Doc Go to Doc#